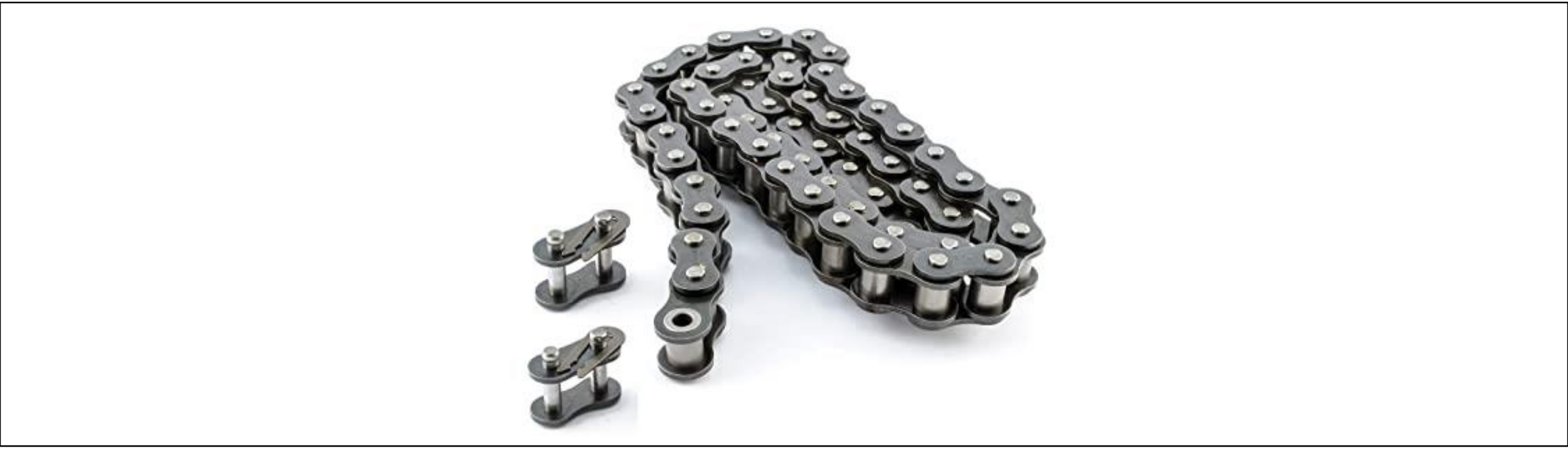
Les chaînes de caractères

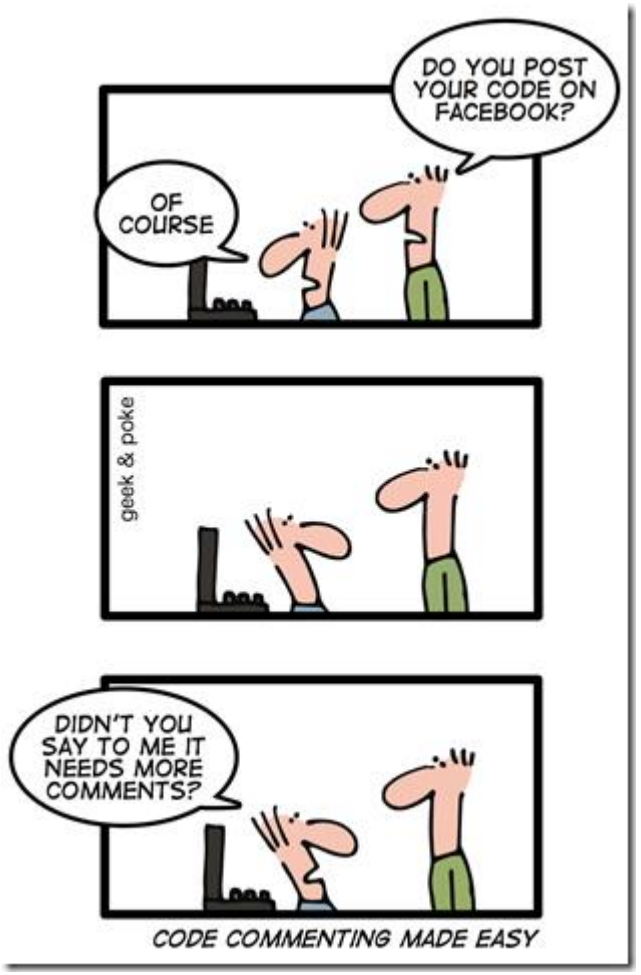


Antoine SCHMID, CFPT-Informatique
Avec la collaboration du CPLN

Table des Matières

- Résumé de l'épisode précédent

- Les strings
  - Rappel
  - Longueur d'une chaîne
  - Minuscules ⇔ Majuscules
  - Comparer deux chaînes
- String = ensemble de caractères
  - Indices
  - Parcourir une chaîne
  - Autres opérations



Dans l’épisode précédent…

- ```
  let films = [
    "Django Unchained",
    "Interstellar",
    "Green Book : Sur les routes du sud"
  ];
  ```

- `films.length`
- `films[0], films[films.length-1]`
- `for, forEach, for-of`
- `push, unshift, pop, shift, splice`
  ```

Rappels sur les chaînes de caractères

- Une valeur de type chaîne (**string**) permet de représenter un texte.
- En JavaScript, on définit une chaîne en plaçant un texte entre guillemets simples (**'je suis une chaîne'**) ou entre guillemets doubles **("je suis une chaîne"**).
- On peut insérer dans une chaîne certains caractères spéciaux en utilisant le caractère **** ("antislash" ou "backslash") suivi d'un autre caractère. Par exemple, **\n** définit un retour à la ligne.
- Appliqué à des chaînes de caractères, l'opérateur **+** déclenche la **concaténation** (jointure) des deux valeurs.

Obtenir la longueur d’une chaîne

- La longueur d'une chaîne (c'est-à-dire le nombre de caractères qui la composent, ou sa taille), est données par la propriété **`length`**.
- On obtient la longueur sous la forme d'une valeur entière.

```
console.log("Je suis une chaîne".length); // 18

// 2ème exemple ou le résultat est stocké dans une variable
let mot = "Kangourou";
let longueurMot = mot.length;
console.log(longueurMot); // 9
```

Convertir une chaîne en minuscules ou en majuscules

```
let motInitial = "Bora-Bora";
console.log(motInitial.toLowerCase()); // bora-bora
console.log(motInitial.toUpperCase()); // BORA-BORA
console.log(motInitial); // Bora-Bora
```

- Il est essentiel de comprendre que la chaîne initiale n'est pas modifiée par ces méthodes
- Toutes les opérations applicables aux chaînes de caractères ne modifient **JAMAIS** la chaîne initiale, mais renvoient de nouvelles chaînes.
- Une fois créée, une chaîne de caractères JavaScript ne peut plus être modifiée. On dit qu'elle est **immuable** (en anglais : immutable).

Comparer deux chaînes

- La comparaison entre deux chaînes s'effectue avec l'opérateur **===**.
- Cette opération renvoie **true** si les deux chaînes sont égales, **false** sinon.

```
const chaine = "qwerty";
console.log(chaine === "qwerty"); // true
console.log(chaine === "azerty"); // false
```

Comparer deux chaînes

- Attention : la comparaison entre chaînes est **sensible à la casse** (case sensitive).

- Une solution pour contrer ceci est la conversion en majuscules ou en minuscules

```
const valeurSaisie = "Quitter";
console.log(valeurSaisie === "quitter"); // false (Q majuscule)
console.log(valeurSaisie.toLowerCase() === "quitter"); // true !
```

String = ensemble de caractères

Associer un caractère à son indice

- Une chaîne de caractères peut être considérée comme un ensemble de caractères.
- Chaque caractère est identifié par un numéro, appelé son **indice** (index en anglais)

- Comme pour les tableaux, l'indice du premier caractère est **0**

Accéder à un caractère par son indice

- Pour accéder à un caractère d'une chaîne, on ajoute **[]** à la chaîne, en indiquant entre les crochets l'indice du caractère
- **maChaine[monIndice]**

```
const sport = "Tennis-ballon";
console.log(sport[0]); // "T"
console.log(sport[6]); // "-"
console.log(sport[13]); // undefined
```

Parcourir une chaîne

- Nous pouvons utiliser une boucle pour accéder successivement à chaque caractère
- Comme le nombre de tour est prévisible, nous choisissons la boucle **for**

```
for (let i = 0; i < maChaine.length; i++) {
  // maChaine[i] renvoie le ième caractère de maChaine
  // ...
}
```

Parcourir une chaîne

- Comme pour les tableaux, on peut utiliser la récente boucle **for-of**

```
let prenom = "Odile";
for (let i = 0; i < prenom.length; i++) {
  console.log(prenom[i]);
}
```

```
let prenom = "Odile";
for (const lettre of prenom) {
  console.log(lettre);
}
```

Transformer une chaîne en tableau

- La méthode **Array.from()** permet de transformer une chaîne en un véritable tableau
- On peut ensuite le parcourir lettre par lettre avec la méthode **forEach()**

```
prenom = "Odile";

Array.from(prenom).forEach((lettre) => {
  console.log(lettre);
});
```

Rechercher dans une chaîne

- La méthode **indexOf()** prend en paramètre la sous-chaîne recherchée.
- Si cette valeur est présente dans la chaîne, elle renvoie l'indice de sa première occurrence. Sinon, elle renvoie **-1**.

```
const chanson = "Honky Tonk Women";

console.log(chanson.indexOf("Tonk")); // 6
console.log(chanson.indexOf("tonk")); // -1 (minuscule)
```

Rechercher dans une chaîne

- **startsWith()** et **endsWith()** permettent de rechercher une valeur au début ou à la fin de la chaîne.
- Ces deux méthodes renvoient **true** ou **false** selon que la valeur soit trouvée ou non.
- Attention, elles sont sensibles à la casse.

```
const chanson = "Honky Tonk Women";

console.log(chanson.startsWith("Honk")); // true
console.log(chanson.startsWith("honk")); // false
console.log(chanson.startsWith("Tonk")); // false
```

Décomposer une chaîne

- Une chaîne de caractères est parfois composée de plusieurs sous-parties séparées par un caractère particulier (point, tiret, point-virgule, etc…).
- Dans ce cas, on peut obtenir toutes ces parties à l'aide de la méthode **split()**.

```
const listeMois = "Jan,Fev,Mar,Avr,Mai,Jun,Jul,Aou,Sep,Oct,Nov,Dec";
const mois = listeMois.split(",");

console.log(mois[0]); // "Jan"
console.log(mois[11]); // "Dec"
```

Quelques méthodes intéressantes

- charAt() Retourne le caractère à la position spécifiée
- charCodeAt() Retourne le code ASCII du nième caractère de chaine
- concat() Retourne la concaténation de 2 chaînes de caractères
- fromCharCode() Crée une chaîne à partir d'une série de code ASCII
- lastIndexOf() Retourne l'indice de la dernière occurrence de sous-chaîne trouvée
- match() Vérifie la concordance d'un motif d'expression régulière
- replace() Remplace dans chaine le motif d'expression régulière par texte
- search() Retourne l'indice de la première occurrence de l'expression régulière dans la chaîne
- slice() Extrait une sous-chaîne
- substr() Extrait de chaine la sous-chaîne contenant nb caractères à partir de l'indice debut
- substring() Extrait la sous-chaîne de la chaîne de caractères entre les positions debut et fin
- trim() Retourne la chaîne débarrassée des blancs et espaces inutiles en début et fin de chaîne

Exercices

- Exercices 9.x : **Strings**